AFRL-AFOSR-VA-TR-2016-0211

New Approaches for Very Large-Scale Integer Programming

George Nemhauser
GEORGIA TECH RESEARCH CORPORATION

06/24/2016
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 30-06-2016	**2. REPORT TYPE** AFOSR final report	**3. DATES COVERED** *(From - To)* 01-04-2012 - 31-03-2016

4. TITLE AND SUBTITLE New Approaches for Very Large-Scale Integer Programming	**5a. CONTRACT NUMBER** FA9550-12-1-0151
	5b. GRANT NUMBER FA9550-12-1-0151
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) George Nemhauser	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Georgia Institute of Technology, North Avenue NW, Atlanta GA 30332	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientic Research, 875 North Randolph Street, Arlington, VA., 22203-1768	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFOSR
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The focus of this project is new computational tools for mixed-integer programming (MIP). During the course of this project we have studied and obtained results on the following topics.

1. Dual heuristics for integer programs in order to rapidly improve dual bounds.
2. Choosing good branching variables in branch-and-bound algorithms for MIP.
3. Machine Learning in solving MIPs.
4. Parallel Processing in Solving MIPS.

The new algorithms are computational tested and, in many cases, outperform existing algorithms. This research has been presented at several conferences and has and will appear in archival journals.

15. SUBJECT TERMS

integer programming, algorithms, parallel processing, machine learning, heuristics

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
U	U	U	UU	2	George Nemhauser
					19b. TELEPHONE NUMBER *(Include area code)* 404-894-2306

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER. Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

AFOSR Final Report

Grant - FA9550-12-1-0151

New Approaches for Very Large-Scale Integer Programming

P.I. George Nemhauser

Georgia Institute of Technology

The focus of this project is new computational approaches for mixed-integer programming (MIP). During the course of this project we have studied the following topics:

Dual heuristics for solving MIPs. While many heuristics have been developed to improve primal solutions, in linear programming based branch-and-bound algorithms, only cutting planes are used to improve dual (lower) bounds. We design a dual heuristic that incorporates relaxation algorithms within a branch-and-bound algorithm.

Generally, lower bounds can be obtained by solving a relaxation, which is achieved by either enlarging the feasible region, decreasing the value of the objective function over the feasible region or both. All commercial MIP solvers use LP relaxations to obtain lower bounds with cutting planes possibly added to improve lower bounds. While cutting planes can be very effective, after adding a large number of them, the LP relaxations can become more difficult to solve. Another type of relaxation is to keep the integrality constraints and relax complicating constraints by simply dropping, dualizing, or aggregating them, which are called *constraint*, *lagrangian* and *surrogate* relaxations, respectively. The goal is usually to rapidly solve one or a sequence of relaxed problems that are computationally easier to handle.

We denote the constraints that are dropped, dualized or aggregated as *relaxed* and the rest of the constraints as *present* in a relaxation problem. Although larger sets of present-constraints in a relaxation lead to better lower bounds, there is a tradeoff between the number of present-constraints and the ease of solving the relaxation. A large number of present-constraints may lead to relaxations that are harder to solve, while a small number may give poor lower bounds. Therefore, our first goal is to understand the impact of the number of present-constraints on lower bounds obtained from different relaxations.

On the other hand, the lower bounds from relaxations with different sets of present-constraints may vary drastically even for a fixed number of present-constraints. We show that for certain MIP problems with a large number of constraints, using linear relaxation dual variable information to choose present-constraints is effective in obtaining good lower bounds. In addition, the quality of the lower bounds obtained with different relaxation techniques may vary depending on the structure of the problem. We compare the lower bounds obtained from constraint and surrogate relaxations. Furthermore, we develop a lazy relaxation. A relaxed-constraint is called *lazy* if it is allowed to be added back to the formulation when it is found to be violated by a primal feasible solution during the branch and bound algorithm. In this case the lazy-constraint is added back to the present-constraint set. Lazy-constraints can be useful for problems with a huge number of constraints especially when there are constraints that are unlikely to be violated by feasible solutions. When using CPLEX 12.2 as the MIP solver with the lazy-constraints option turned on, the lazy-constraints are only checked each time an integer solution is found.

The constraint and surrogate relaxations relax a fixed set of constraints, which can be a barrier for obtaining good lower bounds if the present-constraints are not chosen properly. The lazy relaxation has the flexibility to modify present-constraints during the B&B algorithm, while still relying on choosing an initial set of present-constraints to perform well.

We develop a new heuristic algorithm that solves relaxations with possibly different sets of present-constraints. This idea is analogous to a class of neighborhood search based heuristic algorithms, where neighborhoods of feasible solutions are explored by fixing some variables to obtain better solutions, while our algorithm relaxes some constraints and solves relaxations to obtain lower bounds. Since fixing variables in a primal problem corresponds to relaxing constraints in its dual problem, we call the scheme of heuristically relaxing constraints a *dual heuristic algorithm*. The dual heuristic algorithm chooses present-constraints using information obtained from linear relaxations of the B&B tree nodes. Thus relaxations with different sets of present-constraints can be explored. Insofar as we know, this is the first heuristic algorithm that uses a neighborhood search idea in the dual space.

We conduct our experiments on the *multidimensional knapsack* problem MKP, which is a generalization of the knapsack problem with multiple knapsack constraints. For many MKP instances with a large number of constraints, it is difficult to close the relative optimality gaps even with state-of-the-art MIP solvers. A main obstacle for efficiently solving MKP is the lack of structure. Since all MKP constraints are knapsack constraints, it is difficult to determine effective relaxations in order to obtain good lower bounds. For MKPs with hundreds or thousands of constraints, adding many cutting planes may cause computational difficulties with branch-and-cut algorithms. These characteristics of large size MKPs call for the development of general-purpose lower-bound-improving strategies, which is the focus of this work.

This work was main part of the Ph.D dissertation of Yaxian Li and was published in Y. Li, O. Ergun and G.Nemhauser, A Dual Heuristic for Mixed Integer Programming, *Operations Research Letters* 43, 411-417, 2015.

<u>Choosing good branching variables in branch-and-bound algorithms and machine learning for solving MIPs.</u> A significant aspect of our branching study is to gain a more fundamental understanding of branching through the development and analysis of theoretical models. MIP solvers depend on branching rules to implicitly search the solution space. Numerous experimental results provide a good notion of their performances. However, little literature has been dedicated to theoretical results on MIP branching We design a family of increasingly large IP instances encoding the edge-coloring problem. These instances have trivial feasible solutions. Moreover, we prove that there exists for each instance (of arbitrary size) a fixed-size branch-and-bound (B&B) tree that proves optimality for these solutions. We then give experimental results showing that, to prove optimality, current MIP solvers need an amount of resources that increases steeply with the instance size. Finally, we explain this behavior for SCIP, a state-of-the-art open-source MIP solver.

This work was part of the project on branching of postdoctoral student Pierre Le Bodic and was published in P. Le Bodic and G. Nemhauser, How important are branching decisions: Fooling MIP solvers, *Operations Research Letters* 43, 273-278, 2015.

The selection of branching variables is a key component of branch-and-bound algorithms for solving MIPs since the quality of the selection procedure is likely to have a significant effect on the size of the enumeration tree. State-of-the-art procedures base the selection of variables on their "LP gains", which is the dual bound improvement obtained after branching on a variable. There are various ways of selecting variables depending on their LP gains. However, all methods are evaluated empirically. In this work we present a theoretical model for the selection of branching variables. It is based upon an abstraction of MIPs to a simpler setting in which it is possible to analytically evaluate the dual bound improvement of choosing a given variable. We then discuss how the analytical results can be used to choose branching variables for MIPs, and we give experimental results that demonstrate the effectiveness of the method on MIPLIB 2010 "tree" instances where we achieve a 13% and 16% average time and node improvement, respectively, over the default rule of SCIP, a state-of-the-art MIP solver.

This work was part of the project on branching of postdoctoral student Pierre Le Bodic. It took nearly three years to complete and we consider it to be a significant breakthrough in the theoretical understanding of the branching component of branch-and-bound algorithms for solving MIPs. It has been submitted for publication and is available in *Optimization Online*.

Recently, discrete optimization has been successfully lever-aged to improve machine learning (ML) methodology. We will focus on the opposite direction of this fruitful cross-fertilization. We explore ways to harness ML approaches to improve the performance of branch-and-bound search for MIP. ML techniques have been successfully applied to a number of combinatorial search problems. In the context of MIP, some recent works propose ML techniques for constructing a portfolio of good parameter configurations for a MIP solver, and selecting the best configuration for a given instance.

Variable selection for branching is considered to be a main component of modern MIP solvers. Traditional branching strategies fall into two main classes: Strong Branching (SB) approaches exhaustively test variables at each node, and choose the best one with respect to closing the gap between the best bound and the current best feasible solution value. SB can result in 65% fewer search tree nodes on average, compared to the compared to the state-of-the-art "hybrid branching" strategy. However, this comes at an increase of up to 44% in computation time, as more time is spent per node. On the other hand, Pseudocost (PC) branching strategies are engineered to imitate SB using a fraction of the computational effort, typically achieving a good trade-off between number of nodes and total time to solve a MIP. The design of such PC-based strategies has mostly been based on human intuition and extensive engineering, requiring significant manual tuning (initialization, statistical tests, tie-breaking, etc.). While that approach is important and constructive, we depart from it and propose to learn branching strategies directly from data.

We develop a novel framework for data-driven, on-the-fly design of variable selection strategies. By leveraging research in supervised ranking, we aim to produce strategies that gather the best of all properties: 1) using a small number of search nodes, approaching the good performance of SB, 2) maintaining a low computation footprint as in PC, and 3) selecting variables adaptively based on the properties of the given instance. In the context of a single branch-and-bound search, in a first phase, we observe the decisions made by SB, and collect: features that characterize variables at each node of the tree, and labels that discriminate among candidate branching variables. In a second phase, we learn an easy-to-evaluate surrogate function that mimics SB, by solving a learning-to-rank problem common in ML, with the collected data being used for training. In a third phase, the learned ranking function is used for branching.

Compared to recent machine learning methods for node and variable selection in MIP, our approach: 1) can be applied to instances on-the-fly, without an upfront offline training phase on a large set of instances, and 2) consists of solving a ranking problem, as opposed to regression or classification, which are less appropriate for variable selection. Its on-the-fly nature has the benefit of being instance-specific and of continuing the branch-and-bound seamlessly, without losing work when switching between learning and prediction. The ranking formulation is natural for variable selection, since the reference strategy (SB) effectively ranks variables at a node by a score, and picks the top-ranked variable, i.e. the score itself is not important.

We give an instantiation of this framework using CPLEX, a state-of-the-art commercial MIP solver. We use a set of static and dynamic features computed for each candidate variable at a node, and learning to estimate a two-level ranking of good and bad variables based on SB scores. Experiments on benchmark instances indicate that our method produces significantly smaller search trees than PC-based heuristics, and is competitive with CPLEX's default strategy in terms of number of nodes.

This work is part of the PhD dissertation of Elias Khalil and has appeared in the refereed conference proceedings E. Khahil, P. Le Bodic, G. Nemhauser, L. Song and B. Dilinka, Learning to Branch in Mixed Integer Programming, *American Assoc. Artificial Intelligence Proceedings* 30, 724-731, 2016. An extended version is being prepared for journal publication.

Parallel processing for solving MIPs. Another aspect of this project concerns using parallel processing to solve MIPs. We develop exact algorithms that share information among multiple search trees. We also parallelize local search and feasibility heuristics.

We propose a framework using multiple branch-and-bound trees to solve MILPs while allowing them to share information in a parallel execution. We present computational results on instances from MIPLIB 2010 illustrating the benefits of this framework.

MIP solvers have a multitude of options and the different choices of these options can very significantly affect the performance of the algorithm. Moreover, the performance of a specific implementation on a particular problem instance can vary very significantly with less understood factors, such as the computational environment, random seeds used in the inner workings of the implementation, and permutation of the rows and columns of the instance. One way of exploiting performance variability in solving an instance is processing it with multiple different settings called *configurations* and then selecting the best of these executions.
We study a possible way of exploiting performance variability by considering a diverse set of configurations and executing these in parallel while allowing them to share information among each other. We test different types of information to be shared and compare the performance with that of the base solver with default settings. Our experimental results confirm that by simply selecting the best of multiple runs with different configurations can yield significant performance improvements. We also show that the addition of communication yields substantial benefits in terms of reaching good feasible solutions or good upper bounds quickly. Our approach allows for communication on-the-fly and parallelism is a core part of our implementation.
This work is part of the PhD dissertation of Rodolfo Carvajal and was published in R. Carvajal, S. Ahmed, G. Nemhauser, K. Furman, V. Goel and Y. Shao, Using diversification, communication and parallelism to solve mixed-integer linear programs, Operations Research Letters 42, 186-189, 2014.

Large neighborhood local search is a powerful method for obtaining good feasible solutions to MIPs. Parallelization of large neighborhood search provides the advantage of being able to search multiple neighborhoods simultaneously and to combine the results of individual searches to define more promising neighborhoods. We present a parallel local search approach for obtaining high quality solutions to the Fixed Charge Multicommodity Network Flow problem (FCMNF). The approach proceeds by improving a given feasible solution by solving restricted instances of the problem where flows of certain commodities are fixed to those in the solution while the other commodities are locally optimized. We derive multiple independent local search neighborhoods from an arc-based mixed integer programming (MIP) formulation of the problem. which are then solved in parallel. Our scalable parallel implementation takes advantage of the hybrid memory architecture in modern platforms and the success of MIP solvers in solving small problems instances. Computational experiments on FCMNF instances from the literature demonstrate the competitiveness of our approach against state of the art MIP solvers and other heuristic methods.
This work is part of the PhD dissertation of Lluis Munguia and will be published as L. Munguia, S. Ahmed, D. Bader, G. Nemhauser, V. Goel and Y. Shao, A Parallel Local Search Framework for Fixed-Charge Multicommodity Flow Problems, to appear in *Computers and Operations Research*. It is also available in *Optimization Online*.

The research on parallel large neighborhood search for FCMNF has been extended to finding high quality primal solutions for general MIPs. The approach simultaneously solves a large number of small MIPs with the dual objective of reducing infeasibility and optimizing with respect to the original objective. Both goals are achieved by solving restricted versions of two auxiliary MIPs, where subsets of the variables are fixed. In contrast to prior approaches, ours does not require a starting feasible solution. We leverage parallelism to perform multiple searches simultaneously, with the objective of increasing the performance of our heuristic. Comprehensive computational experiments show the efficiency of our approach as a standalone primal heuristic and when used in conjunction with an exact algorithm.
This work is part of the PhD dissertation of Lluis Munguia and is submitted for publication as L. Munguia, S. Ahmed, D. Bader, G. Nemhauser and Y. Shao, Alternating Criteria Search: A Parallel Large Neighborhood Search Algorithm for Mixed Integer Programs. It also appears in *Optimization Online.*

The feasibility pump is a well-known heuristic for finding a first feasible solution to a MIP. We enhance the basic feasibility pump algorithm by presenting a learning framework that makes use of information collected during the iterations of the feasibility pump. Such information consists of fractional infeasible solutions and their respective final roundings. The framework is able to combine the roundings that are used in subsequent feasibility pump calls. In addition, the framework also provides valuable information that can be used for variable fixing which reduces the size of LP relaxations that need to be solved. We use the feasibility pump heuristic coupled to a biased random-key genetic algorithm (BRKGA). The feasibility pump heuristic attempts to find a feasible solution to a MIP by first rounding a solution to the linear programming (LP) relaxation to an integer (but not necessarily feasible) solution and then projecting it to a feasible solution to the LP relaxation. The BRKGA is able to build a pool of projected and rounded but not necessarily feasible solutions and to combine them using information from previous projections. This information is also used to fix variables during the process to speed up the solution of the LP relaxations, and to reduce the problem size in enumeration phases. Experimental results show that this approach is able to find feasible solutions for instances where the original feasibility pump or a commercial mixed integer programming solver often fail. This work was the project of postdoctoral fellow Carlos Andrade. It has been submitted for publication as C. Andrade, S. Ahmed, G. Nemhauser and Y. Shao, A Learning Framework for the Feasibility Pump.

1.

1. Report Type

Final Report

Primary Contact E-mail
Contact email if there is a problem with the report.

george.nemhauser@isye.gatech.edu

Primary Contact Phone Number
Contact phone number if there is a problem with the report

404-274-5329

Organization / Institution name

Georgia Institute of Technology

Grant/Contract Title
The full title of the funded effort.

New Approaches for Very Large-Scale Integer Programming

Grant/Contract Number
AFOSR assigned control number. It must begin with "FA9550" or "F49620" or "FA2386".

FA9550-12-1-0151

Principal Investigator Name
The full name of the principal investigator on the grant or contract.

George Nemhauser

Program Manager
The AFOSR Program Manager currently assigned to the award

Jean-Luc Cambier

Reporting Period Start Date

04/01/12

Reporting Period End Date

03/31/2016

Abstract

The focus of this project is new computational approaches for integer programming. During the course of this project we have studied the following topics:
• Dual heuristics for integer programs in order to rapidly improve dual bounds. While many heuristics have been developed to improve primal solutions, in linear programming based branch-and-bound algorithms, only cutting planes are used to improve dual bounds. We design a dual heuristic that incorporates relaxation algorithms within a branch-and-bound algorithm.
• Choosing good branching variables in branch-and-bound algorithms for mixed-integer programming (MIP). A significant aspect of our branching study is to gain a more fundamental understanding of branching through the development and analysis of theoretical models. While until now, branching rules have only been evaluated empirically, our new abstract model of branching provides an analytic understanding of branching choices. These analytic results are then translated into a practical algorithm that yields performance exceeding that of classic algorithms on a well-known test bed of MIPs.
• Machine Learning in Solving MIPs. We have developed an algorithm that learns how to choose branching variables from the strong branching rule which is very effective but too slow. The learning algorithm mimics the performance of strong branching but is much faster and improves on the results of conventional branch-and-bound algorithms.

• Parallel processing to solve MIPs. Another aspect of this project concerns using parallel processing to solve MIPs. We develop exact algorithms that share information among multiple search trees. We also parallelize local search and feasibility heuristics.

This research has been presented at several conferences and has and will appear in archival journals.

Distribution Statement

This is block 12 on the SF298 form.

Distribution A - Approved for Public Release

Explanation for Distribution Statement

If this is not approved for public release, please provide a short explanation. E.g., contains proprietary information.

SF298 Form

Please attach your SF298 form. A blank SF298 can be found here. Please do not password protect or secure the PDF The maximum file size for an SF298 is 50MB.

Standard Form 298, AFOSR final.pdf

Upload the Report Document. File must be a PDF. Please do not password protect or secure the PDF . The maximum file size for the Report Document is 50MB.

AFOSR Final Report2016.pdf

Upload a Report Document, if any. The maximum file size for the Report Document is 50MB.

Archival Publications (published) during reporting period:

R. Carvajal, S. Ahmed, G. Nemhauser, K. Furman, V. Goel and Y. Shao, Using diversification, communication and parallelism to solve mixed-integer linear programs, Operations Research Letters 42, 186-189, 2014.

K. Aardal and P. Le Bodic, Approximation algorithms for the transportation problem with market choice and related models, Operations Research Letters 42, 549-552, 2014.

P. Le Bodic and G. Nemhauser, How important are branching decisions: Fooling MIP solvers, Operations Research Letters 43, 273-278, 2015.

Y. Li, O. Ergun and G.Nemhauser, A Dual Heuristic for Mixed Integer Programming, Operations Research Letters 43, 411-417, 2015.

E. Khahil, P. Le Bodic, G. Nemhauser, L. Song and B. Dilinka, Learning to Branch in Mixed Integer Programming, American Assoc. Artificial Intelligence Proceedings 30, 724-731, 2016.

L. Munguia, S. Ahmed, D. Bader, G. Nemhauser, V. Goel and Y. Shao, A Parallel Local Search Framework for Fixed-Charge Multicommodity Flow Problems, to appear in Computers and Operations Research.

P. Le Bodic and G. Nemhauser, An Abstract Model for Branching and its Application to Mixed Integer Programming, submitted.

L. Munguia, S. Ahmed, D. Bader, G. Nemhauser and Y. Shao, Alternating Criteria Search: A Parallel Large Neighborhood Search Algorithm for Mixed Integer Programs, submitted.

C. Andrade, S. Ahmed, G. Nemhauser and Y. Shao, A Learning Framework for the Feasibility Pump, submitted.

2. New discoveries, inventions, or patent disclosures:

Do you have any discoveries, inventions, or patent disclosures to report for this period?

No

Please describe and include any notable dates

Do you plan to pursue a claim for personal or organizational intellectual property?

Changes in research objectives (if any):

None

Change in AFOSR Program Manager, if any:

Initially Dr. Donald Hearn, then Dr. Fariba Fahroo, currently Dr. Jean-Luc Cambier.

Extensions granted or milestones slipped, if any:

A no-cost extension to the initial three year project was granted for the period April 1 2015 - March 31 2016.

AFOSR LRIR Number

LRIR Title

Reporting Period

Laboratory Task Manager

Program Officer

Research Objectives

Technical Summary

Funding Summary by Cost Category (by FY, $K)

	Starting FY	FY+1	FY+2
Salary			
Equipment/Facilities			
Supplies			
Total			

Report Document

Report Document - Text Analysis

Report Document - Text Analysis

Appendix Documents

2. Thank You

E-mail user

Jun 19, 2016 13:46:23 Success: Email Sent to: george.nemhauser@isye.gatech.edu